Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

2015

Real-time detection of operational military information in social media

Gillen, Patrick M.

Monterey, California: Naval Postgraduate School

http://hdl.handle.net/10945/47261

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

REAL-TIME DETECTION OF OPERATIONAL MILITARY INFORMATION IN SOCIAL MEDIA

by

Patrick M. Gillen

September 2015

Thesis Advisors: Thomas Otani
Man-Tak Shing

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704–0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704–0188) Washington DC 20503.		
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2015	**3. REPORT TYPE AND DATES COVERED** Master's thesis
4. TITLE AND SUBTITLE REAL-TIME DETECTION OF OPERATIONAL MILITARY INFORMATION IN SOCIAL MEDIA		**5. FUNDING NUMBERS**
6. AUTHOR(S) Gillen, Patrick M.		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943–5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.		
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

Access to information has never been easier and people's eagerness and ability to publish information on social media platforms has never been higher. The growing mountain of information has presented an opportunity and a significant challenge for data scientists. The military in particular can benefit from the ability to use public information to gain an awareness of its current vulnerabilities as well as learning about its adversaries.

This thesis explores methods for collecting public information from social media that may be revealing operational military movements. This research demonstrates that it is possible to train a machine to search for and find military members in social media by using publicly available information distributed by the military. The postings of military members, once identified, can then be ingested and processed in real time, allowing the timely detection of possible military information that had been posted in social media.

14. SUBJECT TERMS Twitter, tweet, social media, machine learning, support vector machine, big data, data mining, Python, text classification			**15. NUMBER OF PAGES** 77
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

i

THIS PAGE INTENTIONALLY LEFT BLANK

REAL-TIME DETECTION OF OPERATIONAL MILITARY INFORMATION IN SOCIAL MEDIA

Patrick M. Gillen
Lieutenant, United States Navy
B.S., United States Naval Academy, 2006
M.S., George Washington University, 2013

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2015

Approved by: Thomas Otani, Ph.D.
                    Thesis Co-Advisor

                    Man-Tak Shing, Ph.D.
                    Thesis Co-Advisor

                    Peter J. Denning, Ph.D.
                    Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Access to information has never been easier and people's eagerness and ability to publish information on social media platforms has never been higher. The growing mountain of information has presented an opportunity and a significant challenge for data scientists. The military in particular can benefit from the ability to use public information to gain an awareness of its current vulnerabilities as well as learning about its adversaries.

This thesis explores methods for collecting public information from social media that may be revealing operational military movements. This research demonstrates that it is possible to train a machine to search for and find military members in social media by using publicly available information distributed by the military. The postings of military members, once identified, can then be ingested and processed in real time, allowing the timely detection of possible military information that had been posted in social media.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

API	Application Program Interface
CAC	Common Access Card
DoD	Department of Defense
FY	Fiscal Year
ID	Identification
JSON	JavaScript Object Notation
REST	Representational State Transfer
SSBN	Ballistic Missile Submarine
SSN	Attack Submarine
SVM	Support Vector Machine
UTC	Coordinated Universal Time

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First, I would like to thank my wife, Amelia, and son, Patrick. This research required a lot of time away from home in the lab or on travel, and your support never wavered. Amelia, you are an amazing wife and the best mother a child could have. You have sacrificed so much for my ambitions—I recognize that and I am eternally grateful. Thank you. I love you.

Also from my family, I would also like to thank my sister, Cate. I am so proud of you for everything you have accomplished. You are one of the most selfless people I have ever met. I am so happy and thankful that you work so hard to be a part of our lives.

I would also like to thank Dr. Lawrence Shattuck and the Institutional Review Board for taking the time to ensure this research was within the institutional research guidelines.

Special thanks goes to my advisors Dr. Tom Otani and Dr. Man-Tak Shing. Your guidance greatly helped this project from the start and your patience with me is commendable. I always learned something new after talking with either of you.

Lastly, I would like to thank Space and Naval Warfare Systems Command (SPAWAR) Pacific for funding this research. From SPAWAR Pacific I would specifically like to thank Ms. Julie Howell for taking the time to schedule center technical exchanges and providing mentorship along the way.

THIS PAGE INTENTIONALLY LEFT BLANK

I.   INTRODUCTION

Technology has allowed the ways in which people communicate with one another to evolve rapidly. Social media platforms present a tempting and readily accessible way for people to share—instantly, without pause for reflection—their experiences, happiness, frustration, and general opinions on their lives and interactions with the world. One such platform, Twitter, allows users to create a profile and share 140-character messages called tweets. These tweets are then viewable by anyone in the world instantly [1]. The motivation for this research was to see if military users could be found in social media, and whether their postings could serve to tip off followers that a military movement will happen in the near future.

During World War II, the idiom "loose lips sink ships" was popularized as a way to remind people that they should not discuss their relatives' military movements with others for fear that this information could slip into the hands of opposing military planners. In reality, it would be difficult to find a case where loose lips actually did sink a ship during that war. Today, however, communication moves at the speed of light, allowing someone on the other side of the planet to react immediately to information on social media, potentially compromising sensitive military operations.

The military is aware of the potential leak of information through social media, but little has been done to stop information from leaking. The biggest hurdle is figuring out how to stop information from leaking and no one seems to have a good answer. Most leaders point towards training and awareness, as shown in Figure 1 as the key to keeping the information off social media. Unfortunately, these approaches have so far been unsuccessful in stopping the information from leaking.

Figure 1.   DOD Information Assurance Training
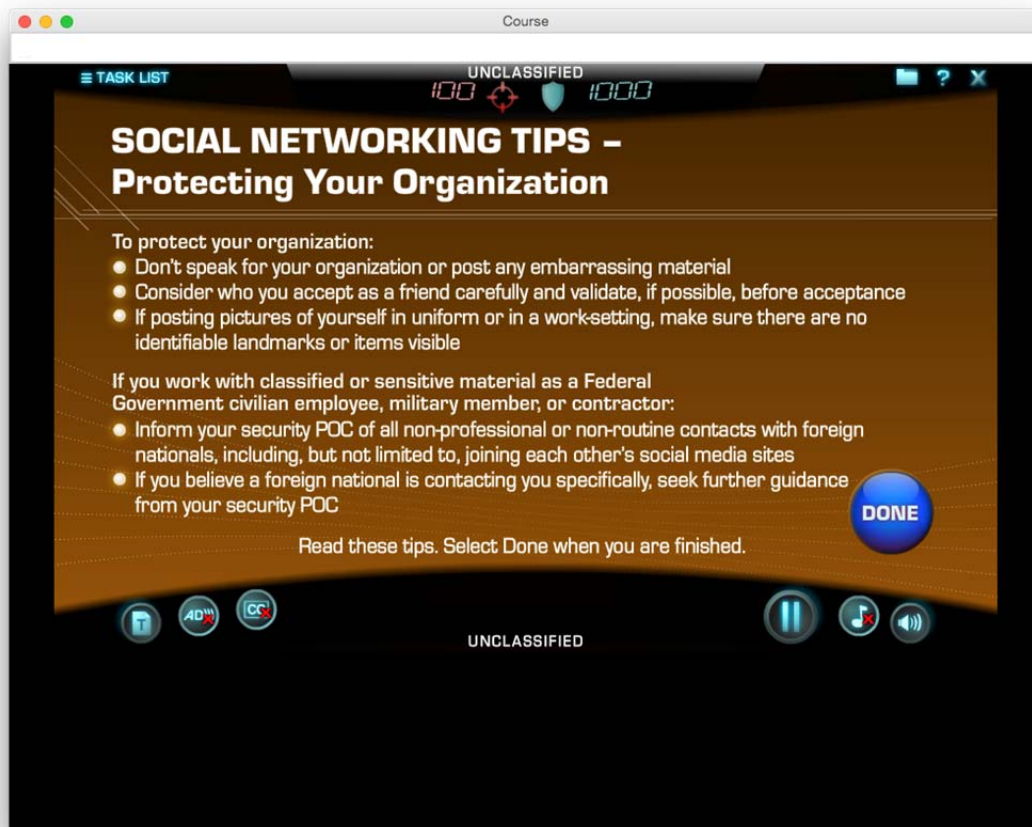
The DoD annual information assurance training gives guidelines to members on
what information should and should not be posted on social networks.

A.   OBJECTIVES

The challenges faced in securing information on social media platforms
also presents an opportunity to the military. The military can benefit from tools
designed to exploit the information posted on social media in the planning
process and to make better-informed decisions based on the information.
Exploiting social media gives operatives a new intelligence resource for which
the cost of entry is very low. The military currently lacks a capability to exploit this
information in real time and integrate it into the current operations and planning
process.

This research explored techniques for finding military members on Twitter based on their profile descriptions and tweets, and to follow these users—alerting the author via a text message when it appeared operational military information had been posted.

B.   EXAMPLES

Before embarking on this study, the author manually searched social media platforms to see how prevalent the leak of operational information was in social media. It was found that members could be found with keyword searches and some of the information that was shared undoubtedly represented operationally relevant and actionable information.

Figures 2, 3, and 4 are examples of an enlisted sailor stationed on a submarine. The chronology goes from newest first to oldest last—exactly as it would appear on Twitter. The gray boxes attempt to hide the identity of the user.

Figure 2.   Enlisted Sailor Twitter Profile
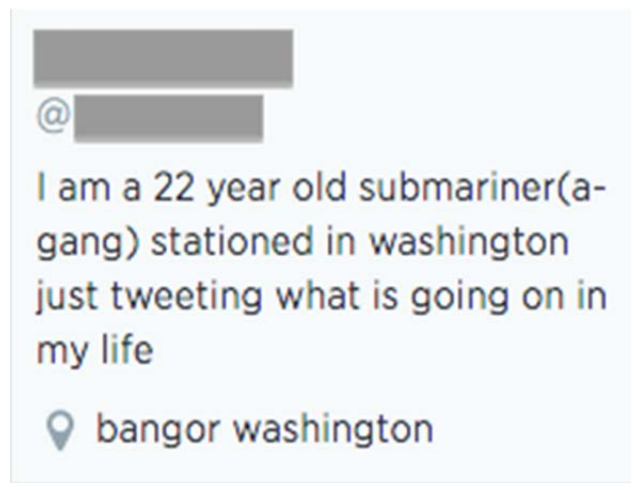
The sailor self identifies as a 22-year-old enlisted member of the auxiliaries division aboard a submarine stationed in Bangor, Washington.

Figure 3.   Sailor Uniform Tweets
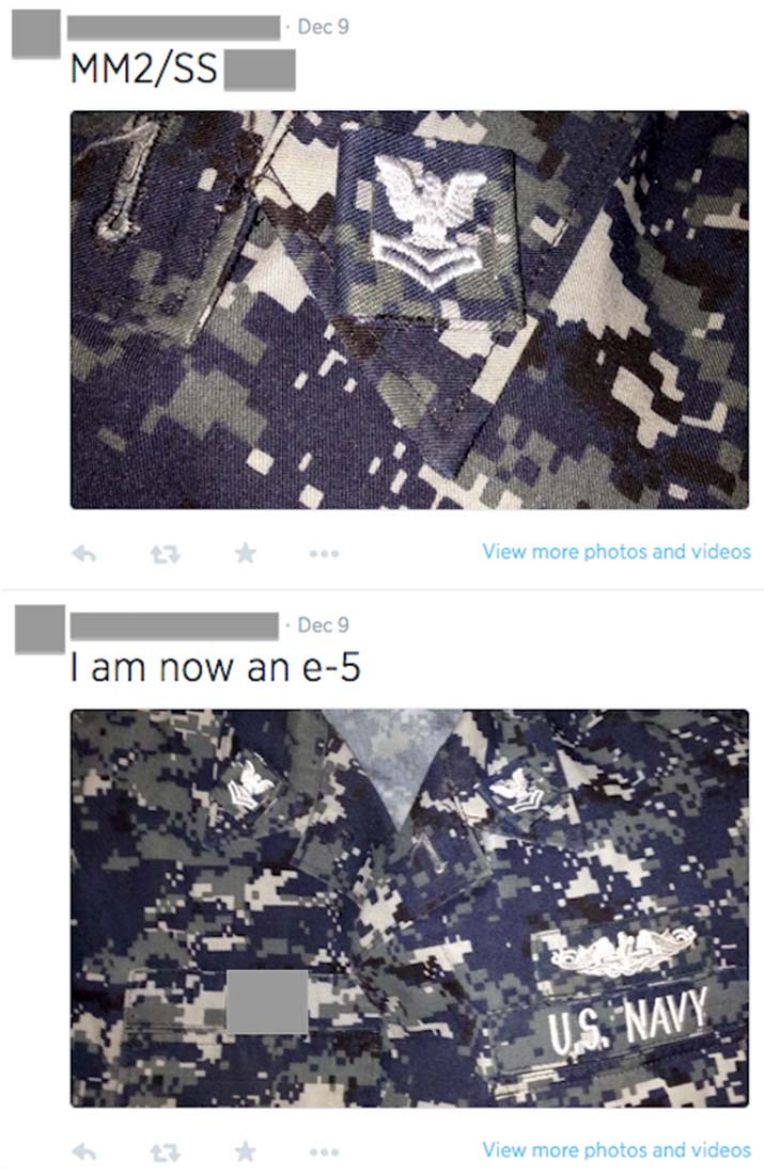
The sailor brags about his recent promotion to Petty Officer Second Class. He shows his uniform with his name, rank, and enlisted submarine warfare insignia reaffirming his profile information.

Figure 4.   Sailor Pre-Deployment Tweets

· Jun 5

Last night of freedom for a couple of months I'll see you at the end of summer

· Jun 5

Last couple of beers before I disappear under the water for a couple of months

View more photos and videos

· May 31

Getting plastered before disappearing for about 4 months
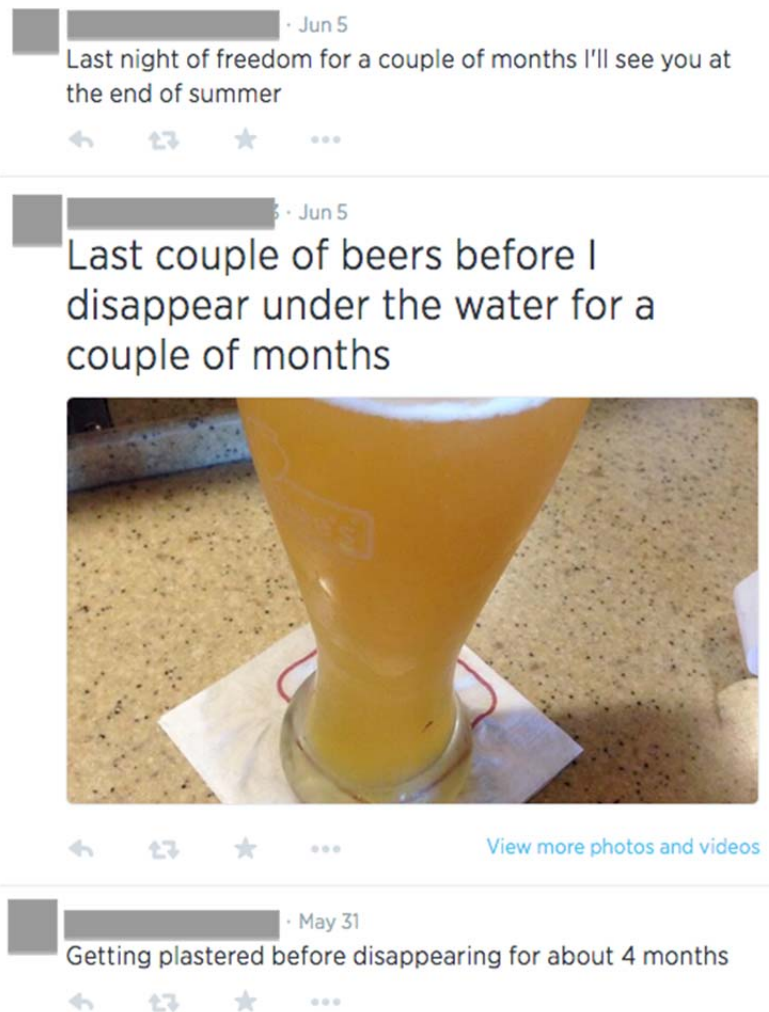
The sailor tweets that he is "disappearing under the water" insinuating that the submarine he is stationed aboard is getting underway. He also uses time specific words enabling one to estimate how long the submarine will be underway: "end of summer," "couple of months," and "about 4 months."

Figure 5.   Sailor Deployment Bookend Tweets

These tweets show the start and conclusion of a deployment. 23 May:
"disappear for a couple months" 29 Aug—3 months 6 days later—"just got back"
He also reiterates that he was on a submarine for this deployment

Figures 6, 7, and 8 are not case studies of a particular user, but rather single examples that demonstrate the kind of exploitable information that can be found on social media platforms. The examples show that officers and enlisted both leak information along with their spouses.

Figure 6.   Facebook Submarine Sighting

This example is from Facebook and is a post from a U.S. Navy officer who is also a private pilot. While flying around Hawaii, he took these pictures of a submarine underway. He identifies it as a Los Angeles Class attack submarine (688) and gives its approximate location and heading. In the comments section, it is also revealed that this hull is the *USS Houston* (SSN-713).

Figure 7.   Sailor Girlfriend Tweets
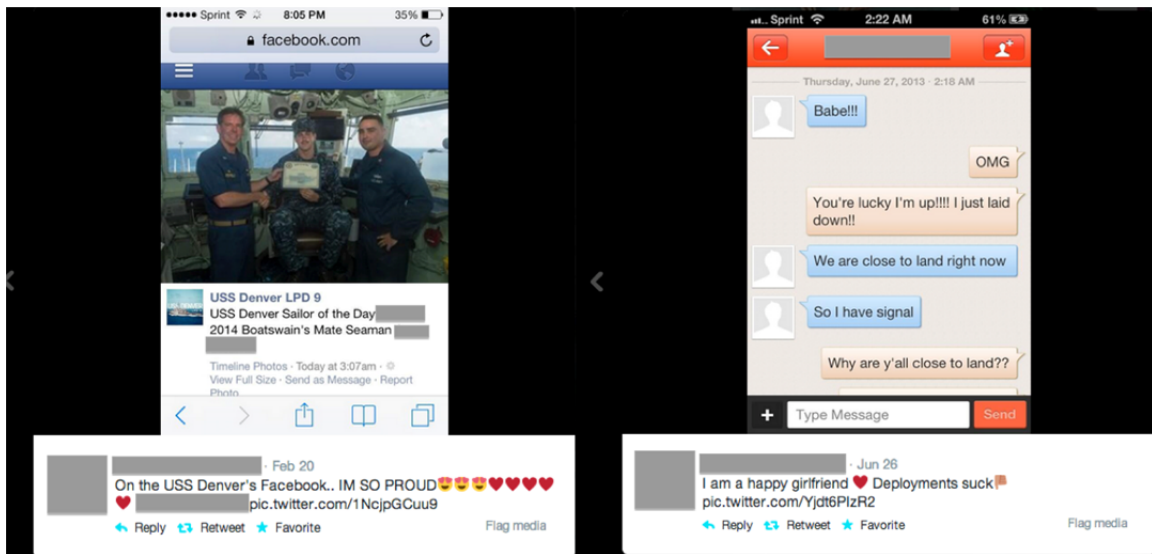
The two posts shown are tweets by a girlfriend of an enlisted sailor. She took screenshots from other applications and posted them on Twitter with her own comments. In February, she posted the screenshot on the left of her boyfriend receiving a sailor of the day award aboard his ship. This post gives his name, rank, and ship. To the right is a screenshot of a private conversation they were having on an instant messaging application. Here, he discloses "we are close to land right now." While this seems innocent enough, this ship is forward deployed to Asia, which leaves a small footprint in which it can be located.

Figure 8.   U.S. Navy Officer Twitter Profile
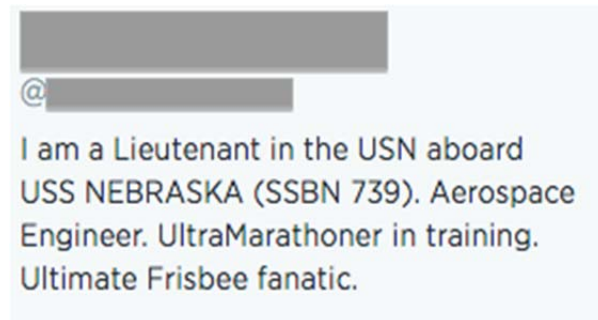
This profile is from a Twitter user. This user self identifies as a U.S. Navy Lieutenant—an officer stationed aboard the *USS Nebraska* (SSBN – 739). The Nebraska is a ballistic missile submarine that conducts the Nation's most secretive deployments. This example shows that officers are also candidates for posting operationally sensitive information in social media.

These previous examples show that relevant military operations information is available on social media. The research presents a technique to detect operational military information in social media. It presents methods and software for finding military users on social media and detecting social media postings from these users that may possibly include information on imminent military movements. We define "imminent" to be a 96-hour period from receiving the information and "military movements" as unit operations wherein the unit departs, enters, or returns to a home or foreign base or port. A detection alert is generated when information indicating a future movement is ingested and processed. The detection is considered successful if the posted information of a user actually contains a general timeframe of the movement, and we can obtain the ship or unit information from this user's profile, other social media postings, or social media relationships. The research will focus looking at the U.S. Navy to determine the best methods and approaches for finding relevant user accounts and ingesting their tweets in real time to alert that operational military information has been leaked in social media. It is expected that a successful algorithm developed here will be applicable towards other domains.

C.   THESIS ORGANIZATION

The research is organized into five chapters. Chapter II presents the background of the research, introduces related work, and explains the technology used in implementing the system. Chapter III describes the specific approach chosen to implement the prototype. Chapter IV discusses the findings of the research. Lastly, Chapter V concludes the thesis and suggests a number of possible extensions of the thesis work.

THIS PAGE INTENTIONALLY LEFT BLANK

II.   BACKGROUND

A.   RELATED WORK

There have been several projects internally and externally to The Naval Postgraduate School (NPS) with respect to social media data and specifically Twitter data. Internally, LT Jeremy Nauta's thesis, *Utilizing Twitter to Locate or Track an Object of Interest*, focused on finding effective methods to utilize the unstructured textual information found in tweets and using that information to find or track a contact of interest [2]. Nauta collected and analyzed about 300,000 tweets [2].

Another NPS thesis written by Kok Wah Ng, titled *The Use of Twitter to Predict the Level of Influenza Activity in the United States* collected several million tweets and attempted to give a better understanding of the influenza patterns in the United States [3]. Ng also attempted to predict when and where influenza outbreaks were likely to occur [3].

Work has also been done in the area of prediction based on Twitter sentiment analysis. [4] focused on ingesting tweets and attempting to predict stock market movement based on the inferred sentiment of the textual information. Research into predicting stock market movement based on Twitter sentiment analysis has thus far been inconclusive, however the lessons learned should prove valuable in this thesis.

The University of Arizona has also done work in the area of tweet tracking and analysis. Their project called *TweetTracker* [5] has the ability to filter tweets based on various user-defined terms. It can then analyze the data using trend analysis and produce a multitude of different visualizations to help the user understand the data [6].

11

B. SOCIAL MEDIA

Social media refers to technology services that allow people to post information about a topic for other people see and interact with. There are many social media platforms—each with their own niche that distinguishes them from the others. Some are focused on videos and pictures while other are geared more toward giving users the ability to create a full online representation of themselves to include likes, dislikes, relationships, and images.

People use social media platforms for a plethora of reasons. Some people use them to stay in touch with friends, others to promote themselves or a product. Many people use social media as their news source for current events. Social media platforms work by allowing users to create the content that others see.

Social media platforms share a common trait that the information posted is generally available to other users around the world instantly. However, various platforms allow users to restrict who can see their information but platforms such as Twitter and Instagram make user's information public by default. Facebook has stricter controls on who can see user data based on user associations between each other. Facebook also requires anyone accessing its platform to have an account.

C. TWITTER ECOSYSTEM

Twitter is a social media company based in California, USA. Their platform allows users to make a profile and post information for others to see and interact with. They charge nothing for the service rather making their income from targeted advertisements that appear on a user's timeline feed along with a product line available to companies and advertisers. The user's timeline feed is populated with posts from accounts that the user follows as shown in Figure 9.

Figure 9.   Twitter Timeline Feed
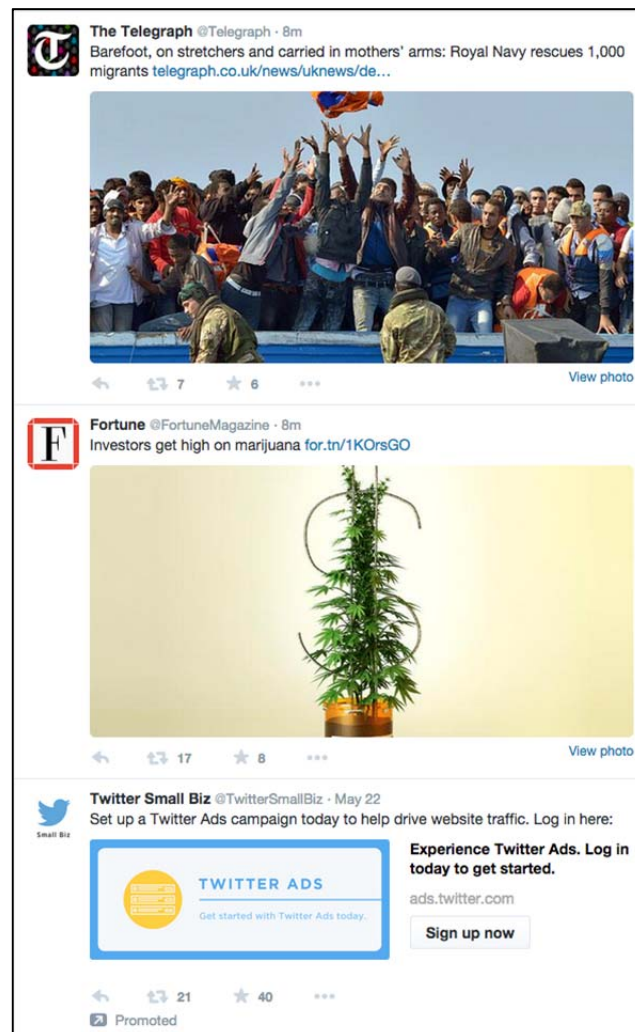
An example of a user's timeline feed. The feed is populated with content from the accounts the user is following. At the bottom is a promoted post. Twitter makes its income by showing users promoted posts from accounts they do not follow.

1.   Why Twitter

Twitter was chosen as the social media platform to collect data for this thesis because it has a very user-friendly application programming interface (API) and because there is no special privilege required to gain access to the information in their databases or their streaming information. The third line of the Twitter privacy policy states, "What you say on the Twitter Services may be viewed all around the world instantly" [7]. Other platforms such as Facebook and

13

LinkedIn would have been great platforms to access and analyze data for this research but their user agreements and API functionality precluded the scale of data mining that would be necessary to meet the goals stated in Chapter I. Facebook and LinkedIn also require significant financing to access mass amounts of user-generated data. Along with not having the financing to buy the data, doing so would have violated the research operating governance set forth by the NPS Institutional Review Board. Without these barriers to entry, the technology approach explored in this thesis is applicable to the data contained in other social media platforms.

2. Twitter Terminology

The Twitter ecosystem has specific terms as detailed in [8] for various types of actions in the system. The research references the following selected terms.

a. User ID

A user identification (ID) is assigned to a user when they sign up for the service. The user ID is unique to the user and cannot be changed. The user ID is a numeric value.

b. Username

The user chooses a username when they sign up for the service. The username is also referred to as the user handle. The handle has to be unique but can be changed an unlimited number of times while the account is active. The handle is comprised of alphanumeric characters and underscores.

c. Tweets

A post on Twitter is called a tweet. A tweet is limited to 140 characters and can contain entities besides text. In addition to text, a tweet can contain links to other webpages and it can have a picture attached to it.

d.   Hashtag

Users also use various textual conventions that are specific to Twitter. A "#" attached to the front of a word is called a hashtag and allows the user to designate their tweet as part of a trending topic or simply to give a one word summary of their point.

e.   @handle

Another convention is placing the "@" symbol at the front of a user's Twitter name to tag them in the post.

f.   ReTweets

The last textual convention is using "RT" plus a username to signify what is being posted is actually a repost (retweet) of another user.

g.   Profile

Users can create profiles for their accounts. The profile can consist of up to 160 characters of text, two pictures and various optional information such as webpage, location, and language. Twitter also attaches the account creation date to the profile. An example of the U.S. Navy's account profile is shown in Figure 10.

Figure 10.   U.S. Navy Twitter Profile

h.   *Friend*

A friend is an account that the user follows on Twitter. The action of friending an account means that the friend account's tweets will populate the user's timeline.

i.   *Follower*

A follower is an account that follows the user on Twitter. To the following account, this action is called friending an account. The follower's timeline will populate with the tweets of the user being followed but nothing will happen to the timeline of the account being followed.

An account has no requirement to friend or follow any accounts. An account can friend or follow another account regardless of whether or not the other account takes any action towards them. This follows with the public nature of Twitter in that anything said on Twitter is publicly available.

j.   *Verified User*

A verified Twitter user is an account that Twitter has confirmed belongs to the user the account claims to be. The anonymity of the Internet allows people to easily pretend to be highly recognizable brands, people, or things. Twitter combats fraudulent accounts by placing a blue checkmark as shown in Figure 10 on the account page so that other users know they are interacting with the authentic account and not a fake account. A verified user typically has many followers, is famous and will not be in the military.

D.   TWITTER API

The scripts used to extract the data from the Twitter database were all built for this research project using Python version 2.7 and the open source Python library called Tweepy.

There are two Twitter APIs available to anyone with a Twitter account. The first API is the streaming API. This API allows the user to access the streaming

data being written by Twitter users in real time. There are several restrictions on the amount of data that a user can obtain for free through the streaming API. First, the user shall apply a filter to the data they are looking for. This filter can be a keyword, username, geolocation, or a combination of these among others. The second restriction is that a nonpaying user can only access up to one percent of the Twitter stream at any given time. Despite these restrictions, a massive amount of data can be had in a short amount of time.

In tests of the streaming API, it was found that with a moderate number of filters in place, one hundred percent of the tweets that met the filter criteria would be displayed to the user regardless of the one percent cap.

Several filters can be placed on the streaming API. First, a keyword filter can be placed on the streaming data. This filter can consist of up to 400 keywords in an "and" or "or" relationship [5]. Next, a geolocation filter can be put on the API. Not all tweets have geolocation but this filter will ensure all tweets returned do have geolocation and plot within a bounding box. Another filter is the filter by user. The streaming API allows a filter of up to 5000 users [5]. This is essentially the same as following these users and allows a program to specifically access the streaming tweets of up to 5000 users without the users knowing their tweets are being collected by the program.

Streaming tweets are received as a JavaScript Object Notation (JSON) object. The JSON object contains a wealth of information not readily visible from the Twitter web interface, including information about the user's profile and the embedded objects in the tweet. An example of a tweet as seen on the web interface is show in Figure 11 and the same tweet viewed through the streaming API is shown in Figure 12. The only filter used to capture the tweet was the word "navy."

Figure 11.   Tweet from Twitter.com Web Interface

Figure 12.   Raw Tweet from Twitter API
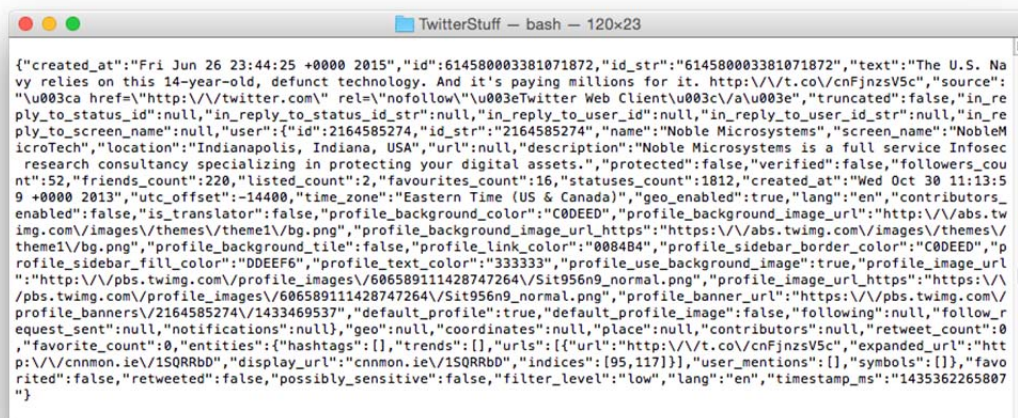
The same tweet from Figure 11 as seen from the Twitter streaming API.

Data through the streaming API can be displayed in various ways. Figure 13 is a display of the streaming API using the same "navy" keyword filter with the time-date stamp and the text of the tweets as they are received.

Figure 13.   Twitter Streaming API Displayed in Terminal

        The second Twitter API allows programmatic interaction with Twitter. Whereas the streaming API is real time receive only, the second API provides the capability to perform POST and GET requests with the Twitter infrastructure. This architecture is called a RESTful architecture or representational state transfer (REST) architecture. [9] The RESTful API is *what* allows third-party applications to post pictures and status updates along with request historical data from the Twitter databases.

        This thesis used both APIs to conduct research. Queries to the Twitter database for users and their historical tweets were performed using the REST API. With users found, the streaming API was used to alert on potential real time and future events.

1.   Rate Limiting

        Twitter has several policies that govern how much data can be obtained from their servers from an application. These restrictions can vary depending on the API being used to access information. The first restriction is the one percent cap on the streaming API. This restriction states that a program can gain access to at most one percent of the real time Twitter feed. This research narrows the stream down to 5,000 users so most tweets make it through despite the restriction. Exceptions happen when there are large social events (New Year's Eve, Super Bowl, etc.) where the 5,000 users may be tweeting so much that

some tweets are not streamed by the Twitter server through the API. However, these instances are rare.

The second access restriction is query rate limiting. This policy is based on the notion that a program has a limited number of requests it can make in a fixed window of time. Rate limiting is fairly common with Internet services and was also seen when accessing the MonkeyLearn API discussed in Chapter III.

Twitter uses 15-minute time windows and limits the number of queries that can be made in this window. The window starts with the first query made and the query limit is based on the information being requested. Table 1 shows the current Twitter rate limit restrictions.

Table 1.   Twitter Rate Limit Restrictions

Title	Resource family	Requests / 15-min window
GET application/rate_limit_status	application	180
GET favorites/list	favorites	15
GET followers/ids	followers	15
GET followers/list	followers	15
GET friends/ids	friends	15
GET friends/list	friends	15
GET friendships/show	friendships	180
GET help/configuration	help	15
GET help/languages	help	15
GET help/privacy	help	15
GET help/tos	help	15
GET lists/list	lists	15
GET lists/members	lists	180
GET lists/members/show	lists	15
GET lists/memberships	lists	15
GET lists/ownerships	lists	15
GET lists/show	lists	15
GET lists/statuses	lists	180
GET lists/subscribers	lists	180
GET lists/subscribers/show	lists	15
GET lists/subscriptions	lists	15
GET search/tweets	search	180
GET statuses/lookup	statuses	180
GET statuses/oembed	statuses	180
GET statuses/retweeters/ids	statuses	15
GET statuses/retweets/:id	statuses	15
GET statuses/show/:id	statuses	180
GET statuses/user_timeline	statuses	180
GET trends/available	trends	15
GET trends/closest	trends	15
GET trends/place	trends	15
GET users/lookup	users	180
GET users/show	users	180
GET users/suggestions	users	15
GET users/suggestions/:slug	users	15
GET users/suggestions/:slug/members	users	15

From Dev.twitter.com. (n.d.). "Rate Limits: Chart | Twitter Developers." (2015).
[Online]. Available: https://dev.twitter.com/rest/public/rate-limits.
Accessed 12 Aug 2015].

Rate limiting presents a hurdle when doing data mining research using Twitter. It is important to ensure every request gets the most copious and high quality data back from the Twitter servers so that no requests are wasted.

E.   MACHINE LEARNING

Machine learning is the premise that a machine—a computer—can score data it has never seen before based on scored data used to train it. Machine learning was used in this research to classify the user profiles and tweets. The current state-of-the-art machine learning model is the Support Vector Machine (SVM).

Broadly speaking, an SVM model attempts to find the best splits in the training data that maximizes their differences and minimizes the crossover between them. With the training data split, it then tries to place new data on one side of the split and give a score to the likelihood that the data is placed on the correct side of the split [10]. A classic graphical representation of this is shown in Figure 14.

Figure 14.   Support Vector Machine Illustration
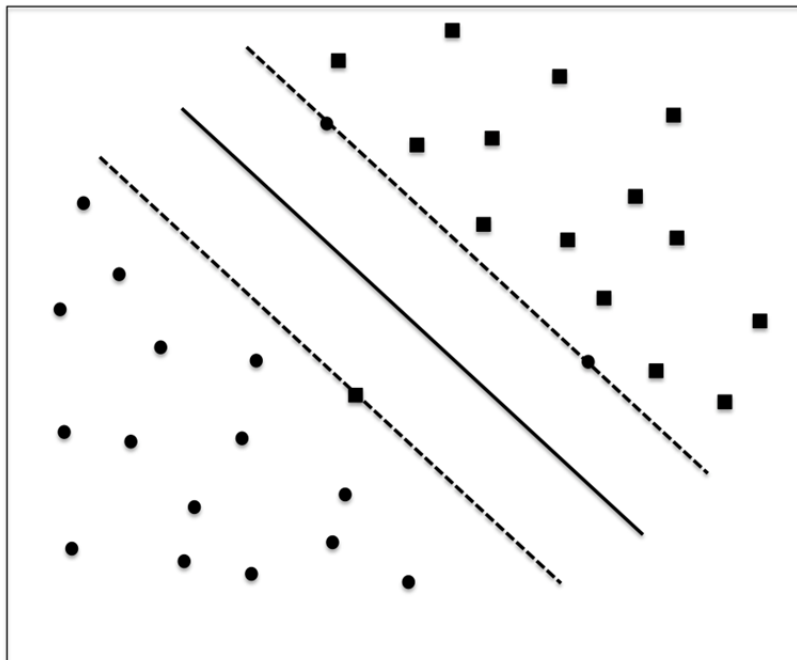
The solid line represents the decision plane and the dashed lines represent the support vectors. After K. Huang, *Machine Learning*. Berlin: Springer, 2008, p. 25.

There are two schools of machine learning—unsupervised and supervised. Unsupervised does not have a training set but rather ingests data and attempts to split the data and group the splits into clusters. Supervised machine learning ingests a known learning dataset and tries to fit new data to the learning set to make a guess as to how the new data should be classified. Figure 15 shows the flow of data to build and use a supervised learning model such as an SVM.

Figure 15.   Supervised Learning Model
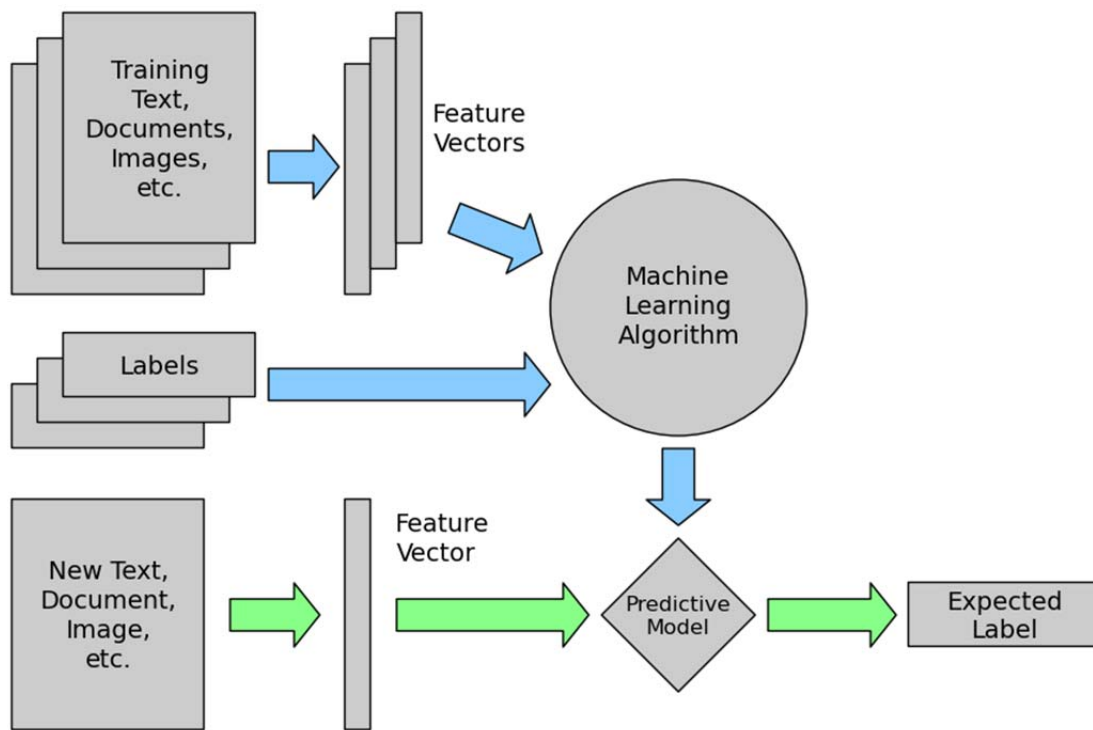
From Astroml.org. (n.d.). "2. Machine learning 101: general concepts — Machine learning for astronomy with scikit-learn." (2015). [Online]. Available: http://www.astroml.org/sklearn_tutorial/general_concepts.html#supervised-learning-model-fit-x-y. Accessed 25 Aug 2015.

F.   UNSTRUCTURED AND NOISY DATA

Mining Twitter and social media generated data in general is very difficult due to what is called unstructured and noisy data. Unstructured data is data that lacks organization. Noisy data is that which contains noise within the data being investigated. User generated data can contain noise in the form of misspellings, extra or missing characters, and needless or excessive punctuation. An example of unstructured and noisy data is the payload of an e-mail. While the e-mail itself has a well understood structure, what the user types in the payload or body of the e-mail has no structure. Twitter has instances of user generated unstructured and noisy data. The first is the user profile description. The only structure in the profile is a 160-character limit. The second instance of user generated unstructured noisy data is the tweet field, which is limited only by 140 characters.

Unstructured and noisy data is difficult for data mining tools. The tools are built using known data and they score the known the data in order to build a model that scores the data being mined. When data is fed into the model, it is processed and scored. Unstructured and noisy data has the tendency to score low because it is very difficult to train the model to handle unstructured and noisy data without the model becoming too large and diluting what is being looked for.

III.   METHOD

A.   APPROACHES

We studied several different approaches for detecting tweets that may contain any operational military information.

The first approach listens to the public Twitter stream and using simple keyword filtering combined with text classification to signal that military operational information had been detected. The advantage of this approach is that it can begin almost immediately if there is already a corpus in place to define the keywords and train the text classifier. The disadvantage is that the streaming API is limited to one percent of the streaming tweets on Twitter.

With the first option, relevant tweets may be captured, but there may be a significant number of tweets that are missed because the streaming API is limited to a cap of one percent of the tweets. Another issue is that the keywords may be too limiting or not limiting enough to catch the relevant tweets. A constant theme with this research was filtering just the right amount of data to be relevant but not overwhelming to a human reviewer.

With the second and third approaches, we first establish a set of users that may be associated with the Navy. With a list of users that appear to be in the Navy, the streaming filter would then become their user IDs. All of their tweets would be ingested and tweets that are classified as operational military information would trigger a detection alert. An aspect that makes this approach attractive is that the built user list has already undergone one level of scrutiny to attempt to validate them as military accounts. When their tweets trigger a detection alert that represents a second level of scrutiny. By the time an alert is issued, the information has been through two levels of scrutiny that attempt to cut down on false alarms—1) the tweet came from an apparent military user, and

2) the tweet contains pieces of text that are classified as military and have a time element to them.

Based on this strategy, a second approach was to look at popular U.S. Navy accounts and analyze their friends and followers lists for accounts of interest. There are a lot of accounts that are associated with the Navy; a few of the most popular ones are shown in Figure 16.

Figure 16.   Popular U.S. Navy Associated Twitter Accounts

		TWEETS	FOLLOWING	FOLLOWERS	FAVORITES
Jonathan W. Greenert ✓	@CNOGreenert	388	213	7,869	11
Naval Forces Europe ✓	@USNavyEurope	5,259	793	11.3K	548
U.S. Navy ✓	@USNavy	18.6K	1,163	489K	109
SECNAV Ray Mabus ✓	@SECNAV	1,513	55	30.1K	17
US FLEET FORCES	@USFLEETFORCES	3,589	104	4,475	1
Destroyer Squadron 7	@DESRON_7	115	184	266	33
PERS-41	@PERS41	450	7	1,811	2

Shown is a sampling of Twitter accounts associated with the U.S. Navy.
Note that the first four are verified accounts annotated by Twitter
with a blue checkmark.

The general U.S. Navy account is the most popular at nearly 500,000 followers. Approach two is advantageous because that among the followers of these accounts, there is probably a high concentration of military users. The disadvantage is that blindly downloading the profiles and tweets of a half million users is very time consuming.

The third approach that was explored and ultimately chosen was using publicly available military promotion lists as the seed for finding users on Twitter.

Specifically, the US Navy promotion lists were chosen. This approach is similar to approach two but the data being downloaded is from directed queries based on confirmed military member names.

The Navy Personnel Command posts the promotion lists of all enlisted sailors ranks E-4 through E-9 and officers ranks O-3 through O-10 on their public website. No sign in or common access card (CAC) privilege is needed to access these lists. The lists are also commonly published on military websites and newspapers upon their release.

B. GATHERING DATA

It is important to note that no special privilege was used to access the Navy personnel lists. There was also no special privilege used to access Twitter. All information downloaded was publicly available, free of charge, and accessed according to their terms of service. The NPS Office of General Counsel and the Institutional Review Board determined that the data being collected by this method was public and therefore not considered to be humans subject research.

The Twitter API ingests a string name as the parameter to search for user profiles. The Twitter server receives the name and returns possible results based on a combination of factors, including profile activity and name match. Twitter also looks for common variations of the name [11]. An example of this methodology is how the Twitter search users algorithm will treat the name "Tom." The search algorithm will also search for "Thom," "Thomas," "Tomas," etc.

Each call to the API user search will return up to twenty possible candidates. The algorithm used for this is shown in the `finder` function in Figure 17. It would make API calls up to five times per name thus having up to one hundred candidates per name.

Figure 17.   Twitter User Search and Tweet Download Python Code

```python
import tweepy
from sets import Set

# keys and secrets are acquired from dev.twitter.com
auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)

# define the API handler
api = tweepy.API(auth)

# input file of names to lookup thru Twitter API
input_names = open("namefile").readlines()

# output file to disk
outfile = open("twitterdata","a")

# list of candidates who have already been downloaded
blacklist = Set([])

# function makes calls to Twitter API to find users given a name
def finder(name):

    # initial API call that returns up to 20 candidates - the max allowed per call
    candidates = api.search_users(q=name,per_page=20,page=1)

    # iterate to build the candidate list up to 100 candidates
    for i in [2,3,4,5]:
        candidates.append(api.search_users(q=name,per_page=20,page=i))

    return candidates

# function to download candidate's tweets
def getTweets(candidate):

    # define list to add tweets to
    all_tweets = []

    # initial API call that returns first 200 tweets - the max allowed per call
    new_tweets = api.user_timeline(screen_name = candidate, count = 200)

    # add new_tweets to all_tweets
    all_tweets.extend(new_tweets)

    # define variable for oldest tweet downloaded
    oldest = all_tweets[-1].id - 1

    # iterate to build the all_tweets list up to 1000 tweets
    while len(all_tweets) < 801:
            new_tweets = api.user_timeline( screen_name = candidate,\
                                            count=200,\
                                            max_id=oldest)

            all_tweets.extend(new_tweets)

            oldest = all_tweets[-1].id - 1

    return all_tweets

# main execution of the program
if __name__ == "__main__":

    for name in input_names:

        candidates = finder(name)

        for candidate in candidates:

            if candidate not in blacklist:
                outfile.write(getTweets(candidate))
```

Note that manipulation of the tweet object before saving is omitted.

28

With the list of candidate user names, the next step was to download the profile and tweet information. The tweet object returns a nested user object that includes profile information so it is not necessary to search for the profile information apart from the tweets.

It is possible to download the most recent 3,250 tweets per user at intervals of 200 tweets per request through the API [12]. A limit of the most recent 1,000 tweets was used. The algorithm to download the user tweets is shown in the getTweets function in Figure 17. The self imposed 1,000 tweet limit was partly due to the Twitter API rate limiting and also a general assumption was used that if a user did not identify as being military in their most recent 1,000 tweets, they probably were not going to be worth following for posting operational military information.

Two simultaneous Python sessions were used to download the user data from Twitter. Each promotion list was about one terabyte of information and the throughput from Twitter was between one and three megabytes per second.

We used the fiscal year (FY) 14 and FY15 E-4 through E-6 promotion lists as the basis for the user search. These two lists collected include nearly 45,000 names, and when passed to Twitter, 1.2 million unique Twitter user identities were returned. Running queries to retrieve the tweets of these users returned approximately 430 million tweets. Chapter IV will discuss the results of the queries in more detail. Data accessed through the API contains objects in JavaScript Object Notation (JSON). The data fields that were saved are shown and described in Table 2.

Table 2.   Twitter Saved Data Fields

Field Name	Description
name_searched_for	The name from the promotion list that returned the unique ID to search
tweet_user_id_searched_for	The user ID that was returned from the promotion list name and passed back to Twitter to download tweets
tweet_created_at_date	The date the tweet was created
tweet_created_at_time	The time (UTC) that the tweet was created
tweet_latitude	The geolocated latitude of the tweet if geolocation is enabled on the account
tweet_longitude	The geolocated longitude of the tweet if geolocation is enabled on the account
tweet_id	The unique ID of the tweet as assigned by Twitter
tweet_favorite_count	The number of times the tweet has been "favorited" by Twitter users
tweet_in_reply_to_screen_name	If the tweet is in reply to another Twitter user, this field populates with the user's screenname
tweet_in_reply_to_status_id	If the tweet is in reply to another Twitter user, this field populates with the user's unique tweet ID
tweet_author_id	The user ID of the author of the tweet
tweet_language	The language of the tweet – self reported by the author
tweet_retweet_count	The number of times the tweet has been retweeted
tweet_source	The source of the tweet – ex. iPhone, Android, Web, etc.
tweet_text	The 140 character text generated by the author and commonly known as "the tweet"
tweet_user_id	The unique user ID of the author - saved as a check to the second field saved. The searched and returned user ID should always be the same
tweet_user_description	The user profile of the author
tweet_user_created_at_date	The date the user account was created
tweet_user_created_at_time	The time (UTC) that the user account was created
tweet_user_followers_count	The number of followers the account has
tweet_user_friends_count	The number friends the account has - this is also known as the accounts this account is following
tweet_user_language	The language of the account as reported by the user.
tweet_user_location	The self reported location of the user
tweet_user_screen_name	The screen name of the user
tweet_user_verified	Boolean value indicating if the user is verified - verified accounts are usually associated with high profile accounts to suppress the prominence of fraudulent accounts
tweet_user_statuses_count	The number of tweets the account has authored
tweet_user_time_zone	The time zone of the user

C. CLASSIFYING USERS

Finding the users manually by reading their profiles and tweets would take too much time to be relevant. In order to speed up the process it was decided to use machine learning to read the data and make determinations of whether or not the profile or tweet was a military user.

Classifying the users was accomplished by utilizing two approaches. The first approach uses the MonkeyLearn online text classification service and the second approach was building classifiers in Python. MonkeyLearn abstracts nearly all of the complication of machine learning from the user. The service allows the user to upload the datasets to train an algorithm and adjust some settings such as classifier type, stop words, n-gram range, and specific options for social media data.

The MonkeyLearn is designed for someone new to machine learning and allows one to begin classifying text almost immediately. MonkeyLearn typically deals with customers that make at most several thousand queries per month. That model did not work for this research because of the volume of data downloaded from Twitter. To overcome this, a dedicated server was provided for one month with unlimited queries. This enabled all the profiles and tweets to be classified in about 23 days through their API.

The MonkeyLearn API can ingest up to 500 texts per request for classification. There is also a rate limit of 30 requests per minute. The script written to interface with the API is shown in Figure 18. The script encoded the texts in JSON format and sent the JSON object as the payload of the API request. The API responds with a payload of the classified texts in JSON format. The response from the API is ingested, decoded from JSON, matched with the user ID of the text and saved to disk.

Figure 18.   MonkeyLearn API Python Interface Code

```python
import requests
import json
from time import sleep
from sets import Set
import csv

# output file
writer = csv.writer(open("classified_profiles.csv","a"))

# set to ensure no profiles are checked twice
checked_ids = Set([])

# function to communicate with monkey learn and save data
def classData(datalist, outputdata):
    try:
        data = {'text_list': datalist}

        response = requests.post(
        "https://api.monkeylearn.com/v2/classifiers/cl_idhere/classify/?",
        data=json.dumps(data),
        headers={'Authorization': 'Token *** unique token ***',
        'Content-Type': 'application/json'})

        results = json.loads(response.text)["result"]

        i = 0
        for result in result:
            writer.writerow(outputdata[i],\
                            str(piece[0]["probability"]),\
                            piece[0]["label"])
            i += 1


    # API is rate limted to 30 requests per minute
    except BaseException, e:
        print e
        sleep(1)
        classData(datalist, outputdata)

# main execution of the program
if __name__ == "__main__":

    # output_data is a list of user ids - this is used
    # when writing the output file
    # data_list is the data sent to the API to classify
    # the two lists have to be seperated while classifying then
    # joined back together when writing the file to disk
    output_data, data_list = [], []

    for line in x:
        p = line.split(",")

        # build list of profiles to classify
        # only check ids that haven't been classified and have text
        if p[0] not in checked_ids and p[1][:-1] != "":
            checked_ids.update([p[0]]) # add the current id to the checked_ids

            output_data.append(p[0]) # build list of current user ids

            data_list.append(p[1]) # build list of profiles

            # API limited to 500 text classifications per request
            if len(data_list) > 499:
                classData(data_list, output_data)
                data_list, output_data = [], []

    # this statement is for the final profiles that didn't add up to 500 total
    if len(data_list) > 0:
        saveData(data_list, output_data)

    # close the file with classification information
    writer.close()
```

32

The second approach was to build our own text classifiers for the user profiles and the user tweets. For this task, a Support Vector Machine (SVM) classifiers were built in Python.

The SVM classifiers built by the author used the Scikit-learn machine learning libraries [13]. The code to build the classifiers is derived from [14] and is shown in Figure 19. The main functions used to build the classifiers include `split_into_lemmas`, the `CountVectorizer`, and the `TfidTransformer`.

Figure 19.   Python Code to Build the Text Classifiers

```python
import csv
from textblob import TextBlob
import pandas
import cPickle
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.pipeline import Pipeline
from sklearn.grid_search import GridSearchCV
from sklearn.cross_validation import train_test_split, StratifiedKFold
from nltk.stem.lancaster import LancasterStemmer

def split_into_lemmas(message):

    # dealing with codecs
    message = str(message)
    message = unicode(message, 'utf-8')
    message = message.encode('unicode_escape').lower()

    # stemming
    st = LancasterStemmer()
    message = message.split(" ")
    wds = []
    for word in message:
        wds = wds + [st.stem(word)]
    message = " ".join(wds)

    words = TextBlob(message).words
    # for each word, take its "base form" = lemma
    return [word.lemma for word in words]

# main execution of the program
if __name__ == "__main__":

    messages = pandas.read_csv('profile_corpus.csv', sep=',', quoting=csv.QUOTE_NONE,
                               names=["message","label"])

    # build training and test sets
    msg_train, msg_test, label_train, label_test = \
        train_test_split(messages['message'], messages['label'], test_size=0.2)

    print len(msg_train), len(msg_test), len(msg_train) + len(msg_test)

    pipeline_svm = Pipeline([
        ('bow', CountVectorizer(analyzer=split_into_lemmas)),
        ('tfidf', TfidfTransformer()),
        ('classifier', SVC(probability=True)),])

    # pipeline parameters to automatically explore and tune
    param_svm = [
      {'classifier__C': [1, 10, 100, 500, 1000], 'classifier__kernel': ['linear']},
      {'classifier__C': [1, 10, 100, 500, 1000], 'classifier__gamma': [0.001, 0.005, 0.0001],\
      'classifier__kernel': ['rbf']},]

    grid_svm = GridSearchCV(
        pipeline_svm,  # pipeline from above
        param_grid=param_svm,  # parameters to tune via cross validation
        refit=True,  # fit using all data, on the best detected classifier
        n_jobs=-1,  # number of cores to use for parallelization; -1 for "all cores"
        scoring='accuracy',  # what score are we optimizing?
        cv=StratifiedKFold(label_train, n_folds = 10),)

    # find the best combination from param_svm
    svm_detector = grid_svm.fit(msg_train, label_train)

    print svm_detector.grid_scores_

    print confusion_matrix(label_test, svm_detector.predict(msg_test))
    print classification_report(label_test, svm_detector.predict(msg_test))

    # store the spam detector to disk after training
    with open('profile_svm.pkl', 'wb') as fileout:
        cPickle.dump(svm_detector, fileout)
```

The `split_into_lemmas` function ingests the training set, tokenizes the words, and stems them. Stemming is the process of striping a word down to its base form [15]. The `CountVectorizer` ingests the result from splitting the words into lemmas and converts the terms into a matrix of token counts [16]. The `TfidTransformer` normalizes the words into the term-frequency inverse document-frequency (TF-IDF) representation. TF-IDF attempts to reduce the impact of words that appear often in the training set because in language, words that naturally appear often are generally not good at classifying the text in which they appear [17]. For example the word "the" appears very often in language but provides little to no value in classifying the text in which it appears.

The program to build the classifier also explores different parameters defined by the user to tune the classifier. The program exploits all logical processing cores on the machine. The research utilized a machine with 24 logical 2.4 GHz processing cores. The profile classifier build took about 4 minutes and the tweet classifier build took about 15 minutes. When complete, the classifier is saved as a Pickle object so that it can be loaded as an object into classification programs without having to be rebuilt.

Using both classification methods allowed for a "best of breed" approach when determining which classification results were more suited for the needs of the research. Combining the results from the profile and tweet classifiers, a list of potential military Twitter users was generated.

D. DETECTION ALGORITHM

Based on the user profile and tweet classification results, the streaming API was accessed using the Twitter user IDs as the only filter. The Twitter streaming API allows a filter of up to 5,000 users per session [5]. Of the 1.2 million users that were originally found, the real time tweets of 30,000 users were ingested. Six streaming sessions were opened and maintained to ingest the tweets of the users.

As tweets were ingested, the words were tokenized and a simple algorithm was used to alert that a movement tweet may have been detected. Tweets that were retweets of other users were ignored—only first person tweets were examined by the algorithm.

Two lists, or buckets, of tokens were used as the basis of the algorithm. The first bucket contained tokens associated with time such as tomorrow, week, month, etc. The second bucket contained military movement type words such as deployment, cruise, underway, etc. If a tweet contained at least one token from each bucket, an alert was issued. Table 3 shows the two buckets of tokens.

Table 3.   Detection Tokens

Navy Tokens		Temporal Tokens	
atlantic	lant	afternoon	schedule
boat	mediterranean	day	scheduled
bridge	navy	days	someday
centcom	ocean	early	soon
cruise	pacific	evening	tardy
deploy	pacom	late	today
deployed	quarterdeck	midnight	tomorrow
deploying	ship	months	week
deployment	submarine	morning	weeks
duty	underway	noon	year
eastpac	watch	now	yesterday
fleet	westpac		
gulf			

The token approach was chosen because there was not enough data available to build a machine classifier that could accurately detect operational military information. With more time, this approach could build a training set for a machine classifier to identify operational military information.

The detection alert would propagate on the terminal screen where all tweets were being displayed and it would also be issued as a text message to the author. This alert method is simple but is platform agnostic and demonstrated

the ease at which the detection alert could be issued cross platform to whatever platform was most convenient to the user—watch center, email, text message, or integration into another application.

Figure 20 shows how the data flows from the user to issuing an alert. Text message was used as the alert mechanism here but it can easily be modified for the appropriate environment.

Figure 20.   Data Flow of Tweet Generation to Detection Alert
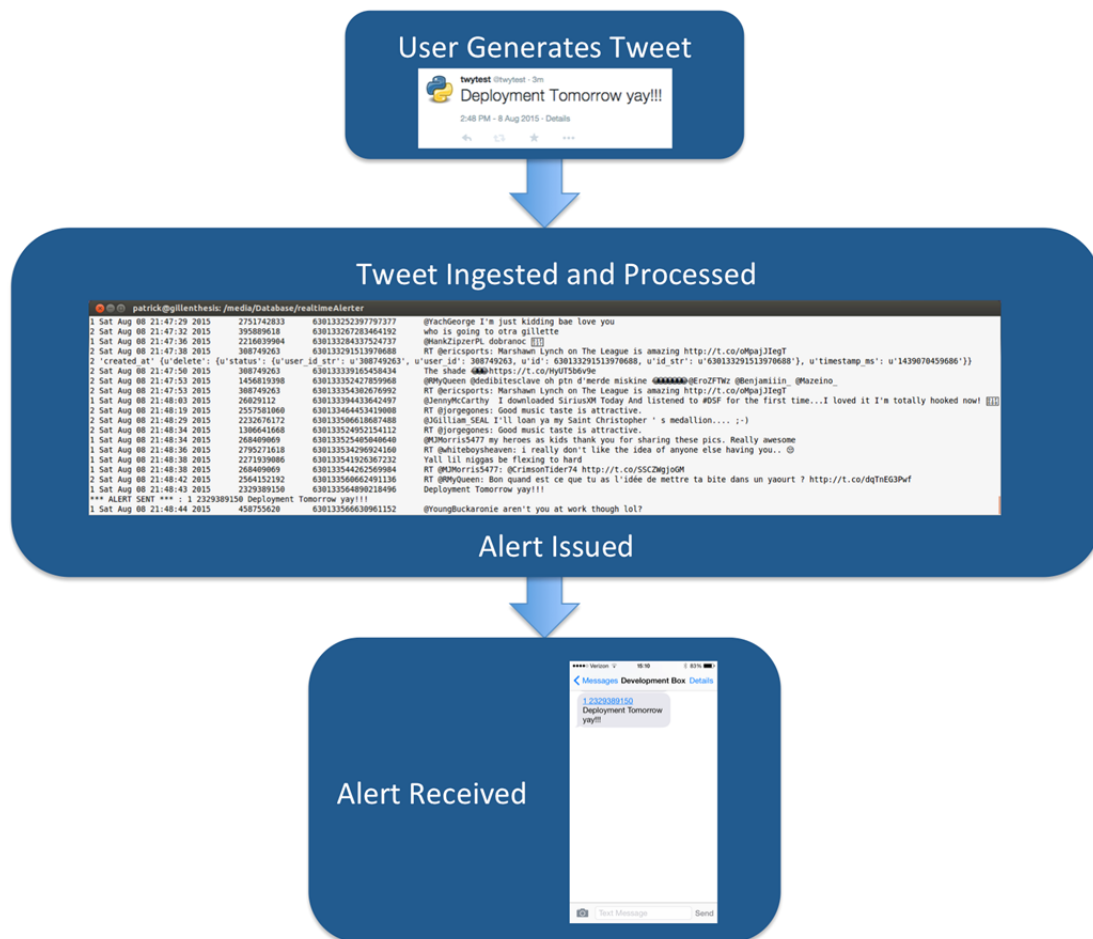
The code to perform the alerting uses the Tweepy Python library to access the streaming Twitter API and is shown in Figure 21. The Tweepy Stream instance connects with the Twitter streaming API and passes messages to a *StreamListener* class instance. Within the class, the *on_data* method receives the tweets [18]. Within *on_data*, the tweet object is decoded from JSON and four data fields are pulled out—the tweet time, the tweet user ID, the tweet ID, and the tweet text. The tweet is written to a file on disk and printed to the screen. Next, if the tweet is not a retweet, the words are transformed to lowercase and tokenized. Tweet token membership is checked against the temporal and military tokens. If the tweet contains at least one temporal token and one military token a detection alert is issued. Here, a text message is sent to the author using simple mail transfer protocol (SMTP).

Figure 21.   Detection Algorithm Python Code

```python
from tweepy import Stream, OAuthHandler
from tweepy.streaming import StreamListener
import json
import smtplib
import csv

# open the file of candidates - 5000 max
candidates = open("candidates.csv").readlines()

# open the file to write out to
writer = csv.writer(open("live_tweets.csv","a"))

# define the temporal word bucket
temporal_words = [  "today","tomorrow","week","month","year","soon","weeks",
                    "afternoon","early","morning","evening","afternoon",
                    "midnight","noon","now","schedule","scheduled","months",
                    "tardy","late","yesterday","someday","day","days"]

# define military word bucket
military_words = [  "navy","ship","duty","watch","underway","submarine","bridge",
                    "deployment","deploy","deployed","cruise","ocean","quarterdeck",
                    "pacific","atlantic","mediterranean","gulf","boat","deploying"]

# alert function
def sendAlert(tweet):
    server = smtplib.SMTP('smtp.gmail.com:587')
    server.ehlo()
    server.starttls()
    server.login('devbox@gmail.com','password')
    server.sendmail('devbox@gmail.com', '2225551212@vtext.com', tweet)
    server.quit()
    print "*** ALERT SENT *** :",tweet
    return

class listener(StreamListener):

    def on_data(self, data):

        # format the data
        data = json.loads(data)

        # pull out data fields and format
        tweet_time      = data["created_at"]
        tweet_user_id   = data["user"]["id_str"]
        tweet_id        = data["id_str"]
        tweet_text      = data["text"]

        # write the data to the output file
        writer.writerow([tweet_time,tweet_user_id,tweet_id,tweet_text])

        # print the data to the screen
        print    proc,tweet_time,"\t",\
                 tweet_user_id,"\t",\
                 tweet_id,"\t",\
                 tweet_text

        # exclude retweets and determine time - military relationship and alert
        if tweet_text[:2] != "RT":
            if any(word in tweet_text.lower().split() for word in temporal_words):
                if any(word in tweet_text.lower().split() for word in military_words):
                    sendAlert(tweet_user_id+" "+tweet_text)

# keys and secrets acquired from dev.twitter.com
auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)

# launch the streaming API and filter based on user IDs
twitterStream = Stream(auth, listener())
twitterStream.filter(follow=ids)

# close the file
outfile.close()
```

THIS PAGE INTENTIONALLY LEFT BLANK

IV.   FINDINGS

The FY14 and FY15 enlisted E-4 through E-6 promotion lists combined consist of about 45,000 names. Queries to Twitter as described in the method section result in about 1.2 million unique profiles and nearly 430 million tweets from these users. The downloading algorithm downloaded about 1 terabyte of data from the Twitter servers for each promotion list. The throughput was between 2 and 3 megabytes per second and downloading the data took about a two months of round-the-clock operations on a dedicated machine.

A.   DATA OVERVIEW

Several interesting statistical data points were produced from the results. These data points are shown in Table 4.

Table 4.   Data Statistics

Names Searched	44,889
Unique Twitter Accounts Searched	1,197,210
Verified Users	5,678
Tweets Downloaded	427,024,296
ReTweets	114,268,219 (26.76%)
Accounts with Profile Data	685,117 (57.23%)
Tweets with Geolocation	13,515,260 (3.17%)
Tweets Geolocated in the US	8,834,450 (2.0%)
Oldest Tweet	13 July 2006
Newest Tweet	24 April 2015

The newest tweet is from the last day of the data download.

Figure 22 gives a pictorial representation of the number of tweets downloaded for a given date. The tweets were downloaded during March and April of 2015. The most recent tweets were downloaded up to 1,000 tweets per user. It is clearly seen that the majority of tweets downloaded were created in the last two years.

Figure 22.   Number of Tweets Created by Date
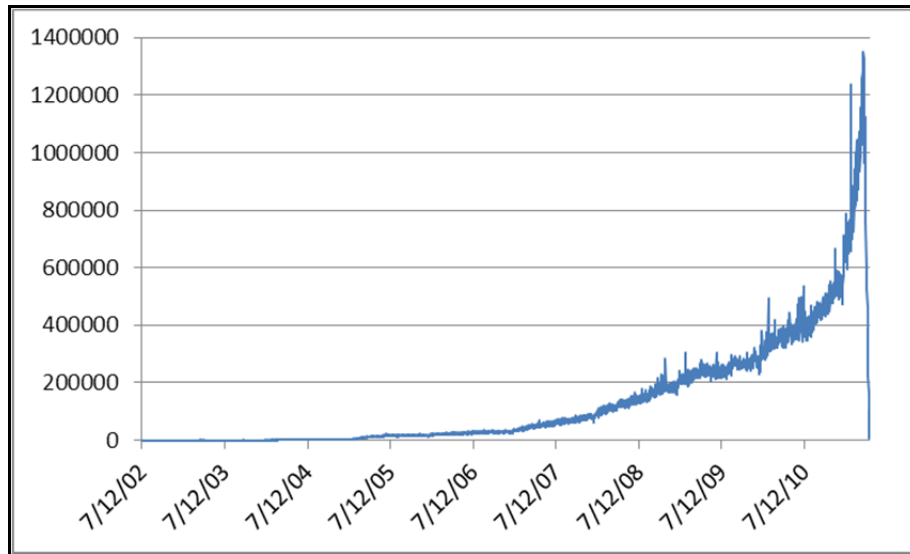
The vertical axis is the number of tweets created and the horizontal axis is the
date. The data point plotted represents the number of tweets created
for a given date.

Figure 23 shows the methods with which users post tweets. This shows
that most users are tweeting using a mobile device such as iPhone or Android.

Figure 23.   Top 20 Sources of Tweets

Source	Count
twitterfeed	1530699
Google	1560968
Mobile Web (M5)	1590448
Twitter for Mac	1808513
Foursquare	1873345
TweetCaster for Android	2076971
Twitter for Windows Phone	2481758
iOS	2648107
Cloudhopper	2770065
Hootsuite	3736137
Echofon	3794675
Twitter for Websites	5651957
TweetDeck	7323434
Facebook	8138987
Twitter for iPad	10971926
Twitter for BlackBerry	11037621
Instagram	12537808
Twitter for Android	65390435
Twitter Web Client	71581629
Twitter for iPhone	171611385

The top twenty tweet sources is a mix of mobile and web based platforms.

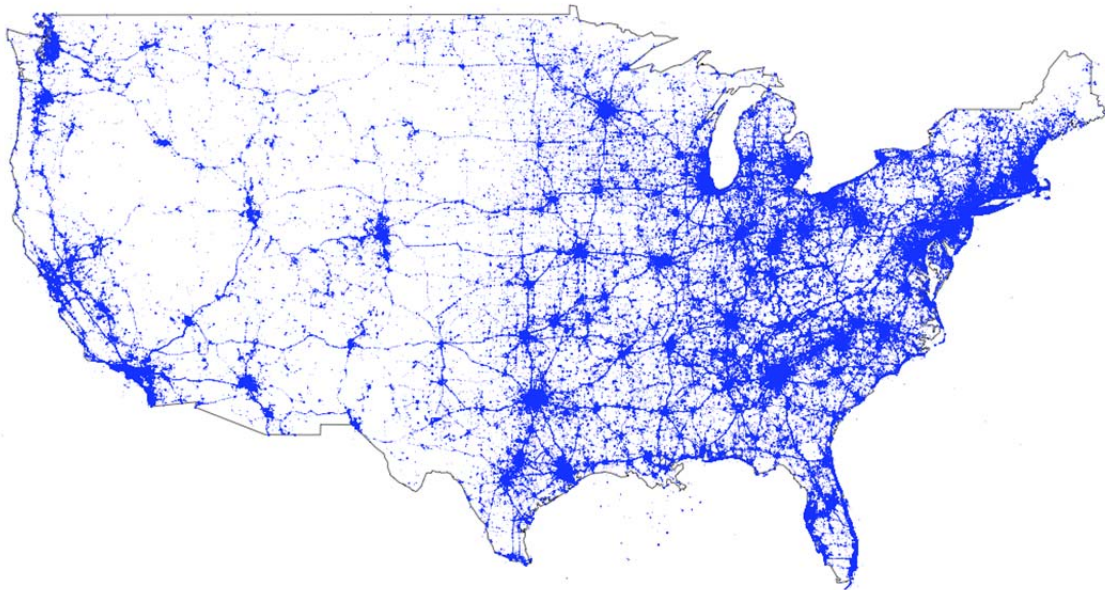Lastly, the geolocated tweets can be plotted on a map as shown in Figure 24. Doing so gives a picture of where people live, work, and travel. With only the continental United States border drawn, the interstate highway system and population centers become clearly visible.

Figure 24.   U.S. Geolocated Tweets Plotted

B.   CLASSIFIER ANALYSIS

The next step after downloading all the data was to attempt to find the military users among the data. The data mining approach was two fold. First, analyze the user profiles for military members and second analyze the tweets for military tweets.

As discussed in Chapter III, Support Vector Machine (SVM) classifiers were chosen to classify the profiles and tweets. The SVM approach required the building of a profile training set and a tweet training set. Building the training sets is a very tedious process but is necessary because it would take too long to manually read all the profiles and tweets.

The training set for the profiles consisted of 207 military profiles and about 6,000 nonmilitary profiles. Finding military profiles manually to train the classifier proved to be a very difficult task. Thousands of profiles had to be manually read to find the 207 actual military profiles. The 6,000 nonmilitary profiles are a combination of validated nonmilitary profiles and the profiles of verified users. The training set is too large to publish, but a word cloud of the training set is shown in Figure 25. Note that the words are stemmed.

Figure 25.   Word Cloud of Profile Training Set
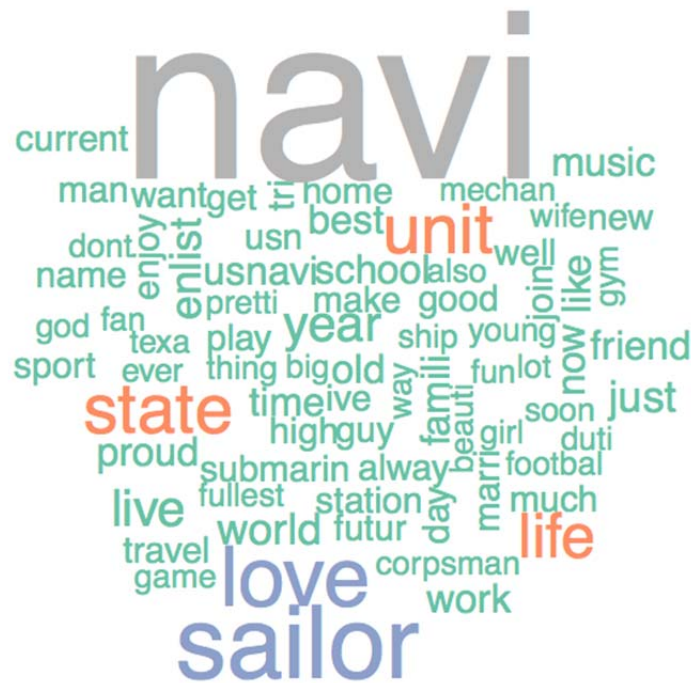
The same training data was used for both the MonkeyLearn classifiers and the classifiers built in Python. When classifying user profiles, there was an 86.5% overlap between the MonkeyLearn and Python classifiers.

A confusion matrix is used to show the accuracy of a classifier and displays the intersection of actual classification by predicted classification. Both classifiers presented similar confusion matrices as shown in Figure 26. It can be

seen that all classifiers had a difficult time accurately differentiating between military and non-military profiles and tweets.

Figure 26.   Classifier Confusion Matrices

Profiles	Predicted	
	Military	Non Military
Actual Military	46	6
Actual Non Military	26	1547

Tweets	Predicted		
	Patriotic	Military	Non Military
Actual Patriotic	441	0	7
Actual Military	0	183	89
Actual Non Military	8	42	1730

Profiles	Predicted	
	Military	Non Military
Actual Military	33	6
Actual Non Military	8	1253

Tweets	Predicted		
	Patriotic	Military	Non Military
Actual Patriotic	389	1	6
Actual Military	2	160	86
Actual Non Military	18	26	1560

C.   DATA ANALYSIS

The tweets of nearly 1.2 million users were downloaded based on the 45,000 names on the two enlisted promotion lists. Of these users, a little more than half (57%) offered some kind of information in their profile. The classifiers identified 8,107 (1.2%) as being military. This is a bit higher than the national average of military to civilians (0.5%) [19], [20] but in line with what was expected.

According to Pew Research, about 1/3 of people aged 18–29 use Twitter [21]. The promotion lists used to find the accounts fall within the 18–29 demographic. We can assume of the profiles without profile data, 1.2% are also military. This results in 6,145 profiles that are military without profile data for a total of 14,252 military profiles among the 1.2 million downloaded. If about 1/3 of our military users in the demographic have a Twitter account, we expect to see about 13,900 accounts among the 45,000 people on the two promotion lists. A word cloud of the found military profiles is shown in Figure 27.

Figure 27.　Word Cloud of Found Military Profiles

The next phase of classification was to classify the tweets the users generated. This was done in an attempt to find the military users who did not provide profile information. SVM classifiers were again used and a training set developed. A word cloud of the training set is shown in Figure 28.

The training set consisted of three categories—military, nonmilitary, and patriotic. Patriotic was chosen as a classification to attempt to help the classifier due to patriotic and military tweets sounding very similar. It was intended that differentiating them in the training set would help the classifier identify the subtleties between them. In total, about 11,000 tweets were used to train the tweet classifier. The confusion matrices for the two classifiers are shown in Figure 26.

Based on the tweet classification results, if the classifier determined that more than 40% of the user's tweets were military, then that user was identified as a potential military candidate. The 40% threshold seems high, but it yields nearly

20,000 users who are unique compared to those identified from their profiles. The tweet classification is not as good as the profile classification so a wider net is required to try to account for the remaining 6,000 users.

Figure 28.   Word Cloud of Tweet Training Set
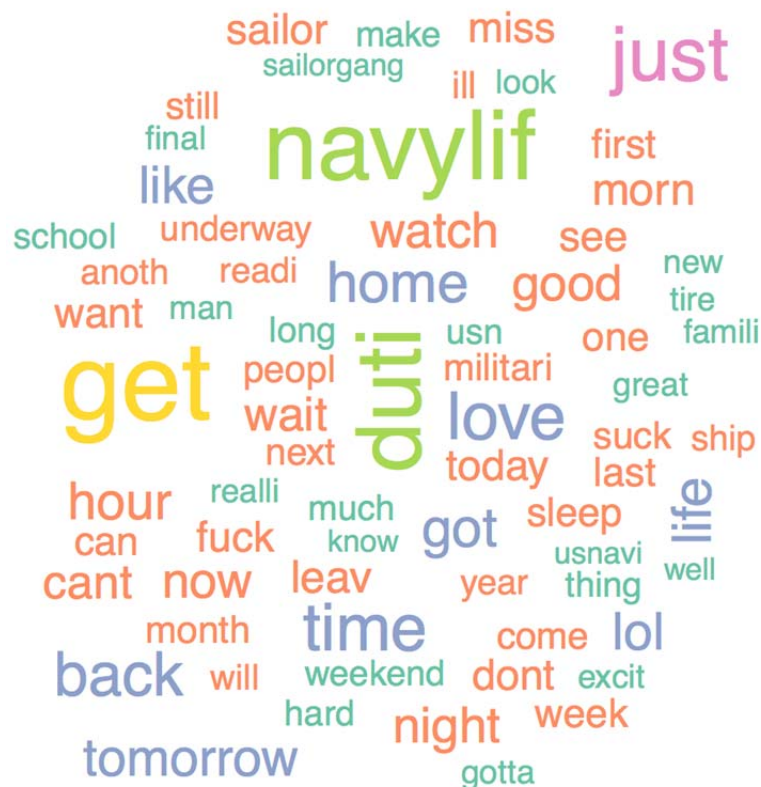
D.   ALERTING

The results of the detection alerting were very interesting. Within two days of launching the alerting service, several alerts were issued that either announced a movement event or confirmed the fact that the user was a military member as shown in Figure 29. Some alerts would confirm both and point to profiles that provided other valuable information about the user, their social connections, and their profession as show in Figure 30. The two buckets of
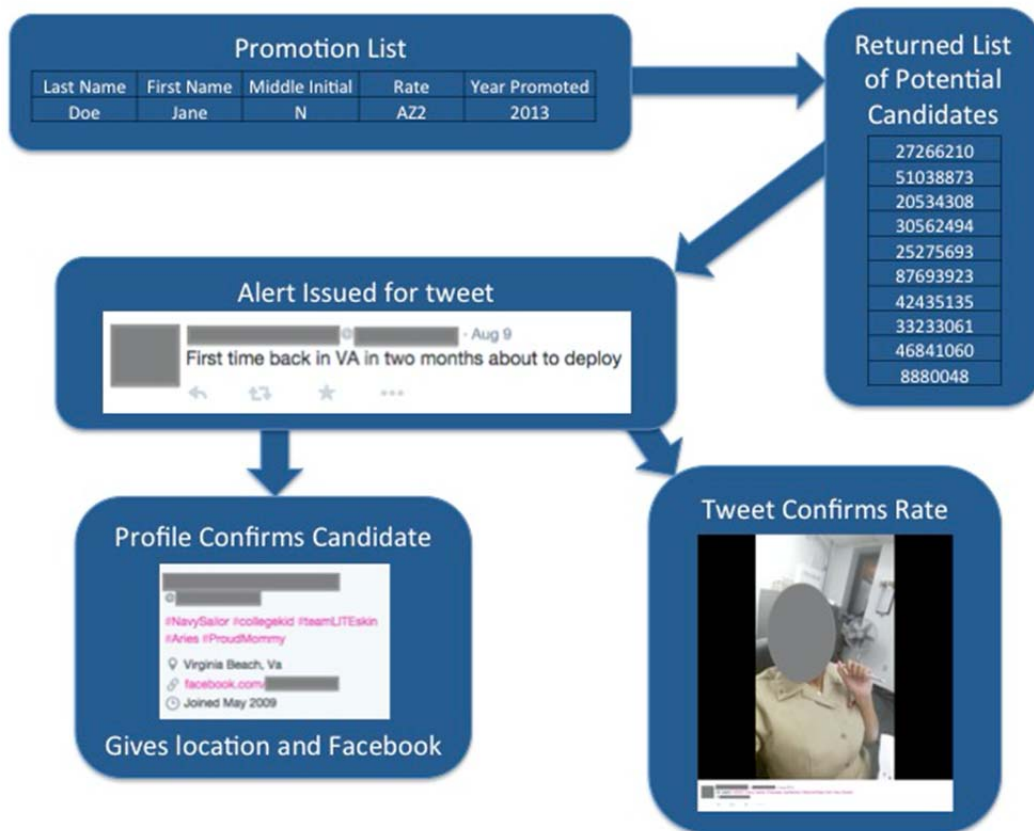
tokens despite being very rudimentary were effective in alerting that something significant was tweeted about.

Figure 29.   Tweets that Issued Alerts

These are examples of tweets that met the criteria for an alert.

Figure 30.   Alerting Process with Candidate Confirmation

This is an example of a user who was being followed based on the ID returned from Twitter from a promotion list name. The user's tweet of an upcoming deployment would not only alert an analyst of the event but also bring attention to the profile to confirm the user is the intended target candidate and point to other accounts the user has such as Facebook and Instagram.

One reason for finding the users first, then using their tweets to alert was to attempt to keep the false alarm rate down. This assumption was proven correct. Along with the streaming sessions that used the user IDs as filters, a streaming session was used that only filtered on the two buckets of tokens and alerted based on the same criteria as the user ID filtered streaming sessions.

The streaming sessions based on user IDs as the filter initially had a false alarm rate of about 50%. Once the tokens were modified to remove "watch" and "now" the false alarm rate dropped to about 20%. The user ID filtered streaming

sessions would issue about five detection alerts per day. The false alarm rate of the streaming session filtered on tokens only was estimated at about 99%. This is an estimate because the token filtered streaming session issued about 500 detection alerts per hour and quickly overwhelmed the user. The first 1,000 detection alerts were read and only two of them proved to be military users.

V.   CONCLUSION AND FUTURE WORK

A.   CONCLUSION

Time was the biggest limitation when conducting this research. The two most time-consuming tasks were collecting the data and classifying the data.

Collecting the data required learning the intricacies of the Twitter APIs including their limitations and boundaries. Downloading the data also took a very long time—nearly two months. This was mostly due to rate limiting, as discussed in Chapter III, and the throughput from the Twitter servers that is available from the public APIs.

Classifying the data also required learning new skills and required a lot of time to read thousands of profiles and tweets to build the training sets. Although the classifiers worked, they could have been better had the training sets been bigger. Their size was limited by not having enough time to manually read more profiles and tweets. The tweet classifier in particular lacked the training data to be highly effective. The machine classifiers also took a fair amount of time to complete the classification. The paid service took 23 days and the locally built classifier took 5 days using 24 processing cores.

This research showed that it is possible using machine learning to automate the discovery of a population of users on Twitter that share a common interest. Here, the common interest was being in the U.S. Navy. With the users found, it was also shown that it is possible to ingest their tweets in real-time and present detection alerts based on combinations of keywords with a low false alarm rate.

B.   FUTURE WORK

The classifiers built for this research worked well, but the noisy and unstructured nature of tweet text data caused many profiles and tweets to be incorrectly classified. The research presented here shows that automated

classification and alerting is possible. The training sets used for this research included hundreds of example target profiles and nearly one thousand example target tweets. Future work should focus on building a large and accurate training set of confirmed profiles and tweets that number into the several thousands.

Future work should also investigate using profile and tweet classifiers against the real-time one percent public stream from the Twitter streaming API. The author attempted this to assess feasibility and it appeared possible. It was found that real-time profile and tweet classification is possible; however, using a single processor for classification was too slow in classifying the stream, causing a backlog and hence not classifying in real time. The API also drops the connection when the backlog gets too big. A project attempting real-time classification of the Twitter stream will need to utilize multiprocessing to keep pace with the tweets as they are transferred by the API to the machine.

The Navy could also benefit from using this research as a foundation to build a tool that unit commanders could use to gauge their unit's exposure on social media platforms. The tool would build a database of user profiles attaching information to the database such as unit, rank, and rate as the user releases it on social media platforms and as it is released officially by the military through mediums such as promotion lists. The tool would then display to unit commanders how much their people are mentioning unit information in the form of a visual tool like a heat map, such as Figure 31. Commanders could then use this tool to make adjustments to their schedule if they feel their operational security has been compromised.

Figure 31.   Example Commander's Schedule Heat Map

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
UNDERWAY	UNDERWAY	UNDERWAY	UNDERWAY	RETURN TO PORT	IN PORT	IN PORT

LEGEND

LOW SOCIAL MEDIA PRESENCE

MODERATE SOCIAL MEDIA PRESENCE

HIGH SOCIAL MEDIA PRESENCE

This figure shows an example of a unit commander's weekly social media presence heat map along with a sample unit schedule for the week. In this example, the commander may want to change the schedule to return to port on Friday because it appears the current schedule has been compromised on social media platforms.

Sentiment analysis is also an emerging area of data science. Commanders may benefit from having a tool that has many of the same characteristics of the tool discussed in the previous paragraph but adds a social media sentiment analysis capability. Knowing unit sentiment would allow commanders to adjust the unit working environment to maximize job satisfaction and productivity.

C.   RECOMMENDATIONS

The military should consider a more secure method of notifying members of their promotion status. The current method of publicly posting the promotion lists on the Internet presents a security risk to the personnel and the units they are assigned to. Besides the examples presented, there were several "high value" personnel found through the method presented in Chapter III. For example, one member found was a sailor who works in the reactor department on an aircraft carrier, and has geolocation enabled on his tweets.

It appears that Navy Personal Command began posting official naval messages online around the year 2000, as shown in Figure 32. Before this, naval messages were sent from Navy Personal Command directly to the units. The messages are unclassified and posting them online is convenient for commanders and sailors but it is also convenient for adversaries. Posting the names of military members openly on the Internet makes finding their social media accounts too easy for nefarious actors.

Figure 32.   Navy Personnel Command Online Messages

Shown is The Navy Personal Command All Navy Messages website. In view are several promotion announcement messages that contain the name and rank of military members. On the left, an archive dating back to 2000 is available.

Military members in the Navy currently access their personnel records, professional data, and training through CAC secured websites hosted by the U.S. government. The Navy should consider designing a capability where each member could be notified of their promotion status on one of these CAC secured

websites. This would be a major course correction for the way Navy Personnel Command conducts their business but the current information age requires that information be treated differently than it has been treated in the past.

Publicly releasing information on the Internet based solely on the fact that it is unclassified is negligent behavior. Information that is unclassified should not warrant a blanket public release on the Internet. Military leadership should assess whether there is a need for the information to be released and weigh that need against the risk of an adversary using that information in a way that could be harmful to the personnel and their units.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1]   Twitter.com. (n.d.). "Privacy policy | Twitter." [Online]. Available: https://twitter.com/privacy. Accessed 19 Feb 2015.

[2]   J. Nauta, "Utilizing Twitter to locate or track an object of interest," M.S. thesis, Sys. Eng. Dept., Naval Postgraduate School, Monterey, CA, 2013.

[3]   K. Ng, "The use of Twitter to predict the level of Influenza activity in the United States," M.S. thesis, Op. Resch. Dept., Naval Postgraduate School, Monterey, CA, 2014.

[4]   A. Porshnex, I. Redkin, and A. Shevchenko, "Machine learning in prediction of stock market indicators based on historical data and data from Twitter sentiment analysis," *Data Mining Workshops (ICDMW), International Conference on*, 2013, pp. 440–444.

[5]   S. Kumar, F. Morstatter, and H. Liu, *Twitter Data Analytics.* New York, NY: Springer Publishing, 2013, pp. 5–72.

[6]   Tweettracker.fulton.asu.edu. "TweetTracker." 2012. [Online]. Available: http://tweettracker.fulton.asu.edu/. Accessed 08 Aug 2015.

[7]   Twitter Inc., Twitter Reports Second Quarter 2015 Results', 2015.

[8]   Twitter.com, "Privacy Policy | Twitter." [Online]. Available: https://twitter.com/privacy. Accessed 19- Feb- 2015.

[8]   Support.twitter.com, (n.d.). "The Twitter glossary | Twitter help center." [Online]. Available: https://support.twitter.com/articles/166337. Accessed 21 Apr 2015.

[9]   J. Sandoval, A. Roussev, and R. Wallace, *RESTful Java web services*. Birmingham, UK: Packt Pub., 2009, pp. 7–79.

[10]   K. Huang, *Machine Learning*. Berlin: Springer, 2008, pp. 1–25.

[11]   Dev.twitter.com, (n.d.). "GET users/search | Twitter Developers." [Online]. Available: https://dev.twitter.com/rest/reference/get/users/search. Accessed 18 Mar 2014.

[12]   Dev.twitter.com, (n.d.). "GET statuses/user_timeline | Twitter Developers." [Online]. Available: https://dev.twitter.com/rest/reference/get/ statuses/user_timeline. Accessed 18 Mar 2014.

[13]   F. Pedregosa, G. Varoquaux, A. Gramfort, and V. Michel, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[14]   R. Rehurek. (2015, Jan.). "Practical data science In Python." [Online]. Available: http://radimrehurek.com/data_science_python/. Accessed 08 Jun 2015.

[15]   Nltk.org, (n.d.). "nltk.stem package — NLTK 3.0 documentation." [Online]. Available: http://www.nltk.org/api/nltk.stem.html. Accessed 08 Jun 2015.

[16]   Scikit-learn.org. (n.d.). "sklearn.feature_extraction.text.CountVectorizer — scikit-learn 0.16.1 documentation." [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.Count Vectorizer.html. Accessed 08 Jun 2015.

[17]   Scikit-learn.org. (n.d). "sklearn.feature_extraction.text.TfidfTransformer — scikit-learn 0.16.1 documentation." [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTr ansformer.html. Accessed 08 Jun 2015.

[18]   Docs.tweepy.org. (2015). "Streaming with Tweepy — Tweepy 3.3.0 documentation." [Online]. Available: http://docs.tweepy.org/en/v3.4.0/streaming_how_to.html. Accessed 26 Aug 2015.

[19]   L. Howden and J. Meyer, "Age and sex composition: 2010," *2010 Census Briefs*, Washington, DC: U.S. Dept. of Commerce, 2011.

[20]   R. Hale, *United States Department of Defense Fiscal Year 2016 Budget Request Overview*. Washington, DC: U.S. Dept. of Defense, 2015.

[21]   M. Duggan, N. Ellison, C. Lampe, A. Lenhart, and M. Madden. (2015). "Demographics of key social networking platforms," *Pew Research Center: Internet, Science & Tech*,. [Online]. Available: http://www.pewinternet.org/2015/01/09/demographics-of-key-social-networking-platforms-2/. Accessed 09 Aug 2015.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
    Ft. Belvoir, Virginia

2. Dudley Knox Library
    Naval Postgraduate School
    Monterey, California